

Ejercicio 5:

5) Para pasar el programa a lenguaje de máquina, primero lo escribiremos en lenguaje ensamblador y luego haremos la traducción.

MOV R0, 0x0002	→	0001 1000 0000 0000 0000 0000 0000 0010	En hexadecimal, 1800 0002 1840 0020 7821
MOV R1, 0x0020	→	0001 1000 0100 0000 0000 0000 0010 0000	
ADD R0, R1	→	0010 1000 0010 0001	

Ejercicio 6:

6) MOV R0, 0x00FF → Ingreso c16: 0000 0000 1111 1111 → MOV R0, 0x00FF. Finalmente, al programa queda:

MOV R1, 0x1000 → Ingreso c16: 0001 0000 0000 0000 → MOV R1, 0x1000

ADD R0, R1 → ADD R0, R1

Finalmente, al programa queda:

- MOV R0, 0x00FF
- MOV R1, 0x1000
- ADD R0, R1

Ejercicio 7:

7) a) MOV R1, 20 → Siendo que el modo de direccionamiento es inmediato: R1 = 0x0020

b) MOV R1, [20] → El modo de direccionamiento es directo: R1 = [0x0020] = 0x0040 (contenido de la celda de dirección 0x0020)

c) MOV R1, [[20]] → El modo de direccionamiento es indirecto: R1 = [[0x0020]] = [0x0040] = 0x0060 (acceso doble a la memoria)

d) MOV R1, R0 → El modo de direccionamiento es de registro: R1 = R0 = 0x0030 (guarda en R1 el contenido de R0)

e) MOV R1, {R0} → El modo es de registro indirecto: R1 = {R0} = [0x0030] = 0x0050 (toma el contenido de R0 como dirección de memoria)

f) MOV R1, {R0+20} → El modo es de registro indirecto: R1 = {R0+20} = [0x0030+0x0020] = [0x0050] = 0x0070 (modo direccionamiento directo con la dirección de memoria formada por la suma entre el contenido de R0 y 0x0020)